Федеральное государственное автономное
образовательное учреждение
высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт космических и информационных технологий
институт

Кафедра «Информатика»
кафедра

УТВЕРЖДАЮ
Заведующий кафедрой

_____ А. С. Кузнецов
подпись инициалы, фамилия

«_____» _____ 2019 г.

БАКАЛАВРСКАЯ РАБОТА

09.03.04 Программная инженерия
Код – наименование направления

Разработка модуля «Организация учебного процесса» для мобильного приложения «Я в СФУ»
тема

Руководитель	_____	доцент, к. т. н.	А. В. Хныкин
	подпись, дата	должность, ученая степень	инициалы, фамилия
Выпускник	_____		И. С. Байкалов
	подпись, дата		инициалы, фамилия
Нормоконтролер	_____	доцент, к. т. н.	О. А. Антамошкин
	подпись, дата	должность, ученая степень	инициалы, фамилия

Красноярск 2019

РЕФЕРАТ

Выпускная квалификационная работа по теме «Разработка модуля «Организация учебного процесса» для мобильного приложения «Я в СФУ» содержит 47 страниц текстового документа, 20 использованных источников, 22 рисунка, 1 приложение.

МОБИЛЬНОЕ ПРИЛОЖЕНИЕ, БАЗА ДАННЫХ, ANDROID, СТУДЕНТ, УЧЕБНЫЙ ПРОЦЕСС, УСПЕВАЕМОСТЬ, ЗАЧЕТНАЯ КНИЖКА, ПРОФИЛЬ СТУДЕНТА, ЗАКАЗ СПРАВОК.

Целью работы является разработка модуля «Организация учебного процесса» для мобильного приложения «Я в СФУ».

Для достижения поставленной цели решались следующие задачи:

1) Проанализирована предметная область.

2) Сделан обзор и анализ существующих решений.

3) Спроектирован и разработан модуль мобильного приложения,

а) разработана база данных;

б) реализована графическая часть клиента;

в) реализована функциональная часть клиента;

г) реализован сервер модуля «Организация учебного процесса» мобильного приложения «Я в СФУ».

4) Проведено тестирование и устранение ошибок в разработанном модуле.

В ходе выполнения данной работы были разработаны структура базы данных и архитектура мобильного приложения. Также был создан сервер и реализован клиент для обработки и отображения полученных данных от сервера.

Было получено свидетельство о государственной регистрации программы для ЭВМ №2019615282 от 23.04.2019 г. В настоящее время идут переговоры о возможности интеграции мобильного приложения «Я в СФУ» в информационное пространство СФУ.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

В наше время, несмотря на значительный рост популярности iOS, Android является ведущей мобильной операционной системой в мире. Удобность системы, привлекательный интерфейс, скорость работы, широкий функционал и доступность устройств – все это увеличивает количество пользователей смартфонов на базе операционной системы Android.

Сейчас практически никто не говорит «мобильный телефон», потому что это уже не только средство связи, но и по-настоящему умный девайс, именуемый в народе «смартфон». Использование социальных сетей, возможность вести бизнес, просматривать веб-страницы, расплачиваться за покупки и многое другое – для нас это стало привычно, и для современного человека остаться без своего смартфона сопоставимо с потерей некоего функционала.

Существует несколько проблем, связанных с сервисами СФУ, из-за которых наше приложение будет иметь актуальность среди студентов. Одна из самых главных – это проблема удобства пользования сервисами, так как сами сервисы плохо адаптированы для мобильных устройств или не адаптированы вовсе. Вместо того, чтобы каждый раз открывать браузер, вводить логин и пароль и искать необходимую информацию, студенту будет достаточно авторизоваться только при первом входе в приложение. Вторая немаловажная проблема – отсутствие необходимой информации под рукой при отсутствии интернета. Наше приложение способно решить эту проблему, так как хранит информацию в локальной базе данных на устройстве пользователя.

В наше время студент живет в бешеном ритме и может забывать, что ему нужно сделать, какое расписание на завтра и многое другое. Разработанное нами приложение позволит каждому студенту СФУ быть в курсе всех новостей своего института, всегда знать актуальное расписание занятий, иметь под рукой удобный ежедневник, следить за своей успеваемостью – и все это с помощью своего смартфона.

Наша команда разработала данное приложение с целью агрегирования различных сервисов СФУ, другими словами, это позволило нам объединить некоторый функционал сервисов в одно приложение для того, чтобы облегчить взаимодействие с ними.

Так как у современного человека смартфон почти весь день находится при нем, студенту, при составлении планов на день, нужно всего лишь взять свой смартфон и узнать расписание занятий или проверить ежедневник на наличие важных дел и тогда он точно ничего не пропустит и не забудет, ведь его смартфон надежно хранит все в локальной базе данных или может получить данные из хранилища удаленного сервера.

На текущий момент не существует единого мобильного приложения, способного в удобном виде предоставить всю необходимую студенту информацию. В магазине приложений Google Play был найден аналог нашего проекта – мобильное приложение «Студент СФУ». Проанализировав аналог, мы пришли к выводу, что данное приложение не решает вышеизложенных проблем, приложение не было доработано и отображает лишь расписание занятий.

В рамках данной дипломной работы будет описано создание единой информационной системы на базе ОС Android, ориентированной на студентов. Оценив всю необходимость создания данного приложения, все плюсы и минусы системы Android, мы пришли к выводу, что приложение обязательно найдет свою аудиторию и будет пользоваться спросом среди студентов Сибирского федерального университета.

В дальнейшем развитие данного приложения не будет прекращено, планируется внедрение новых сервисов и переход в статус официального приложения СФУ.

1 Анализ предметной области

Целью данной работы является разработка мобильного приложения для упрощения использования сервисов Сибирского федерального университета, в частности «Мой СФУ», «Официальный сайт СФУ», «Довузовское управление СФУ» и прочих. В рамках текущей работы предлагается следующее решение поставленной задачи – приложение, в функционал которого входит два модуля:

- студент:
  - авторизация;
  - расписание;
  - успеваемость;
  - зачетная книжка;
  - профиль;
  - моя группа;
  - новости университета и института;
  - ежедневник.
- абитуриент:
  - новости университета и довузовского управления;
  - институты;
  - общежития;
  - подбор направления;
  - контакты приемных комиссий институтов;
  - карта кампуса СФУ.

1.1 Анализ требований целевой аудитории к разрабатываемому программному продукту

Зачастую, разработчики программных продуктов делают большой акцент на процессе программирования, уделяя недостаточно внимания общению с

клиентом. Многие разработчики не умеют или не желают проводить сбор требований, а клиенты зачастую не хотят тратить свое время на разработку технического задания программного продукта. Недостаточный объем информации, поступающей от клиента, требования, сформулированные не полностью, их кардинальное изменение – это главные причины, из-за которых командам разработчиков не удается в срок и в рамках бюджета предоставить клиентам всю запланированную функциональность продукта [1].

Для того, чтобы не было допущено таких ошибок, важно грамотно подойти к управлению требованиями. Управление требованиями – это систематический подход к выявлению, организации и документированию требований к системе [2].

Для сбора требований целевой аудитории к разрабатываемому программному продукту было проведено анкетирование с использованием сервиса Google Forms. Анкета содержала вопрос, касающийся раздела «Студент». Далее мы рассмотрим результаты, которые были получены по итогам анкетирования. На рисунке 1 представлен сводный результат анкетирования в процентном соотношении.

В рамках выполнения данной дипломной работы по разработке модуля «Организация учебного процесса» клиент-серверного мобильного приложения «Я в СФУ» мы рассмотрим требования, полученные от опрошенных студентов в результате анкетирования.

У студентов был один вопрос, который состоял из полей с выбором ответа для того, чтобы сориентировать анкетируемого, так и из полей для самостоятельного заполнения. В результате анкетирования были получены следующие результаты, представленные на рисунке 2.

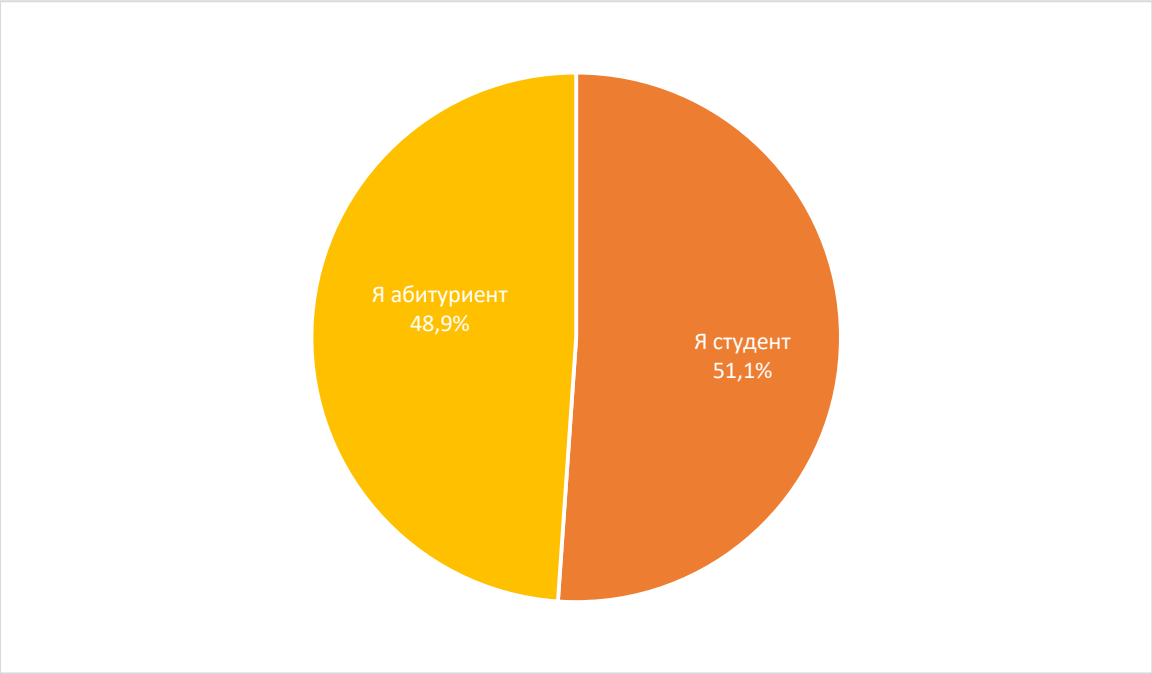


Рисунок 1 – Сводный результат ответов о статусе анкетированного

Рисунок 2 – Результаты анкетирования по вопросу, посвященному основным возможностям модуля «Студент»

Итак, по результатам анкетирования были составлены основные требования к разрабатываемому модулю, а именно:

- модуль должен содержать информацию об успеваемости студента в электронных курсах;

- модуль должен содержать информацию из зачетной книжки и предоставлять ее в удобном виде;

- в модуле должен быть реализован профиль студента, список группы и связь старосты с группой;

- в модуле должна быть реализована возможность заказа справок;

– авторизация пользователя в модуле должна происходить с помощью данных, полученных при поступлении в университет

1.2 Особенности предметной области и решаемых задач

Смартфоны занимают большую часть рынка, об этом можно судить исходя из распределения мирового Web-трафика за 2018 год по различным устройствам [3], представленного на рисунке 3.

Как можно увидеть в графике, больше половины трафика всего мира проходит через смартфоны. Также за 2018 год средняя скорость мобильной передачи данных заметно выросла, а более быстрое интернет-соединение помогает снизить уровень стресса, что следует из публикации компании Ericsson в своем отчете Ericsson Mobility Report [4].

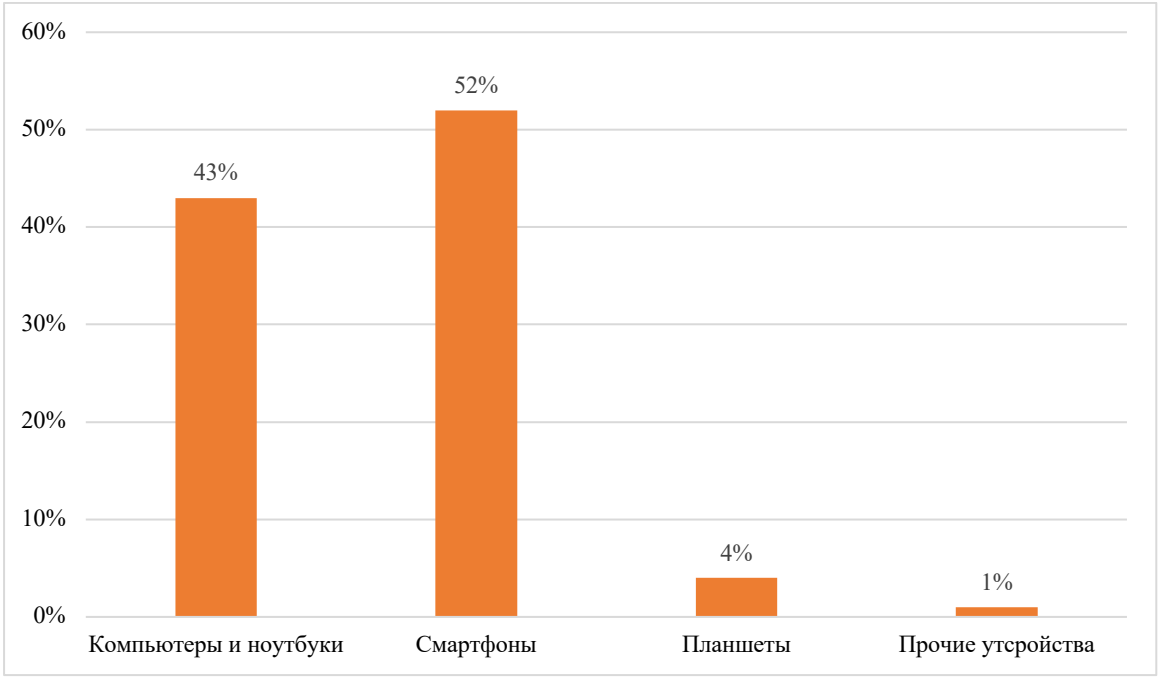


Рисунок 3 – Распределение WEB-трафика по устройствам в 2018 году

Смартфон очень легко можно сопоставить компьютеру, а в некоторых аспектах может и превосходить его. Сейчас уже никого не удивишь смартфоном с мощностями среднего компьютера, при этом размер позволяет брать его всегда с собой, не тратя больших усилий. Это стало одной из главных причин решения о реализации клиент-серверного мобильного приложения для студентов и абитуриентов СФУ. Сайт СФУ и сервис «Мой СФУ» содержат важную информацию, но большинство этой информации не требуется студенту или абитуриенту постоянно, также не всегда удобно заходить на эти ресурсы с персонального компьютера, ведь его иногда просто может не быть под рукой, а адаптивная верстка данных сервисов под мобильный вид отсутствует, что не позволяет использовать данные сервисы с небольшого экрана смартфона.

Из преимуществ смартфона перед персональным компьютером можно выделить такие факторы, как:

- компактность;
- переносимость;
- сравнительно низкая цена;
- удобное сенсорное управление.

Когда есть идея написания мобильного приложения, необходимо принять решение, какую платформу для приложения выбрать. Для того, чтобы сделать выбор, необходимо обратиться к статистической информации по распределению операционных систем проданных смартфонов в России. Согласно сайту сбора глобальной статистики StatCounter [5], на момент сентября 2018 сложилась следующая ситуация, представленная на рисунке 4.

Как видно их гистограммы, на рынке мобильных устройств в России преобладают устройства под управлением Android, в пользу которого и был сделан выбор.

После выбора операционной системы устройства разработки, необходимо решить, какую минимальную версию платформы поддерживать. Для определения минимальной версии обратимся к официальному сайту Android Developers [6]. На рисунке 5 представлены данные с панели распределения версий платформ.

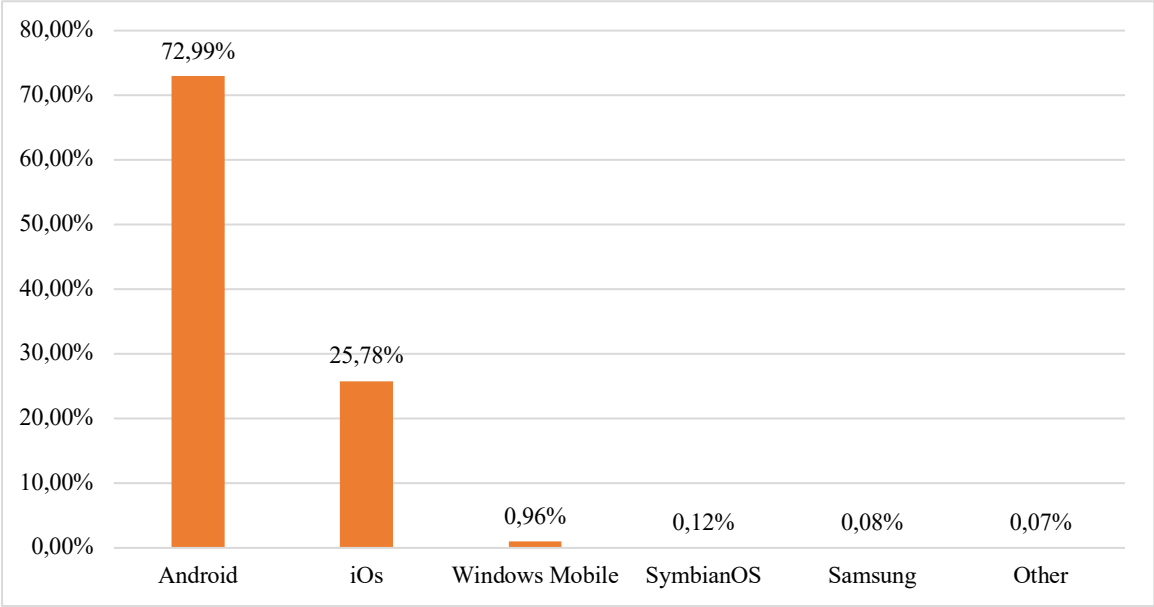


Рисунок 4 – Распределение операционных систем проданных смартфонов в России

Как можно судить из гистограммы, 88,9% пользователей системы Android используют версию платформы 5 и выше. Также, начиная с версии Android 5.0 Lollipop, был введен Material Design, яркий и отзывчивый дизайн пользовательского интерфейса. Исходя из этих данных и из желания дать пользователям наиболее красивый и интерактивный интерфейс, было приято решение поддерживать версию платформы не ниже 5, что соответствует версии API от 21 выше.

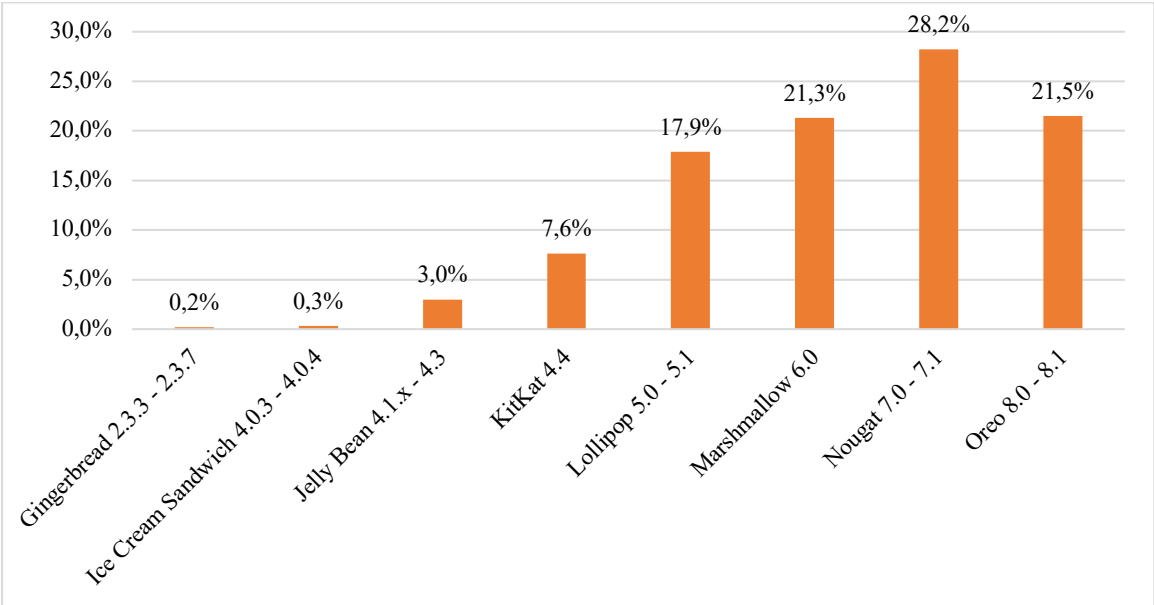


Рисунок 5 – Распределение версий платформ Android

1.3 Выбор средств разработки

1.3.1 Среда разработки

Для нативной разработки мобильных приложений под Android OS, как правило, используют IntelliJ IDEA или Android Studio. Для создания клиентской части мобильного приложения была выбрана среда разработки IntelliJ IDEA, разработанная JetBrains, так как команда разработчиков уже имела лицензию на данное программное обеспечение. Данная среда включает в себя следующие преимущества [7]:

- умное автодополнение, инструменты для анализа качества кода, удобная навигация, расширенные рефакторинги и подсветка синтаксиса используемого языка;

- профессиональный набор инструментов для разработки Android-приложений;

- интеграция с автоматизированными инструментами сборки и управления проектом, включая Maven, Gradle, Ant и другими;

- инструменты для тестирования с поддержкой JUnit, TestNG, Spock, ScalaTest и spec2;

- интеграция с системами управления версиями, включая Git, Subversion, Mercurial и CSV.

Для работы с серверной частью мобильного приложения на Node.js была выбрана IDE WebStorm, также от компании JetBrains. WebStorm [8] – среда для разработки на JavaScript, которая подходит для client-side-разработки, создания приложений на Node.js и мобильных приложений на React Native.

Главное достоинство WebStorm – это удобный и умный редактор для JavaScript, HTML и CSS, который также поддерживает TypeScript, CoffeeScript, Dart, Less, Sass и Stylus и фреймворки, например, Angular, React и Vue.js.

WebStorm, обеспечивая подсветку синтаксиса и автодополнение кода, проверяет его на ошибки, помогает быстро перемещаться по проекту и безопасно

вносить изменения с помощью рефакторинга. В WebStorm есть инструменты для отладки кода и интеграция с системами управления версиями.

Было принято решение использовать IDE одного семейства продуктов для реализации разработанного продукта, за счет их наибольшей совместимости при реализации и тестировании разрабатываемого ПО.

1.3.2 Расширяемый язык разметки XML

Для построения макетов для будущего дизайна мобильного приложения традиционно используется язык XML [9].

Язык XML был создан для хранения, транспортировки и обмена данными, с его помощью можно реализовать обмен данными между различными системами. XML документ состоит из частей, называемых элементами. Элементы составляют основу XML-документов. Они образуют структуры, которые можно обрабатывать программно или с помощью таблиц стилей. Элементы размечают именованные разделы информации. Элементы строятся с помощью тегов разметки, обозначающих имя, начало и конец элемента. Элементы могут быть вложены друг в друга, на верхнем уровне находится элемент, называемый элементом документа или корневым элементом, в котором содержатся остальные элементы.

Документ XML может располагаться в одном или нескольких файлах, причем некоторые из них могут находиться на разных машинах. В XML используется специальная разметка для интеграции содержимого разных файлов в один объект, который можно охарактеризовать как логическую структуру. Благодаря тому, что документ не ограничен одним файлом, XML позволяет создавать документ из частей, которые могут располагаться в любом месте. Документ XML обычно содержит следующие разделы:

- XML-декларация;
- Пролог;
- Элементы;
- Атрибуты.

1.3.3 Объектно-ориентированный язык Java

Выбирая язык, мы смотрели в сторону нативной разработки. Нашим выбором стал язык Java [10].

История Java восходит к 1991 году, когда группа инженеров из компании Sun Microsystems под руководством члена Совета директоров Джеймса Гослинга (James Gosling) занялась разработкой небольшого языка, который можно было бы использовать для программирования бытовых устройств, например, контроллеров для переключения каналов кабельного телевидения. Сейчас Java является самым популярным и востребованным языком программирования в Enterprise проектах, что привело к устойчивому росту популярности и в более маленьких проектах.

Java имеет следующие качества:

- язык Java независим от платформы, на которой выполняются программы: один и тот же код можно запускать под управлением операционных систем MacOS, Windows, Linux, что очень важно, так как работа нас проектом в целом, велась разработчиками на разных платформах;

- полностью объектно-ориентированный язык;

- память в языке Java освобождается автоматически с помощью механизма сборки мусора;

- Java обладает большой библиотекой программ для передачи данных на основе таких протоколов TCP/IP;

- многопоточность.

При разработке модуля язык Java использовался для управления графическими элементами экрана и построения алгоритмов обработки полученных данных с сервера.

1.3.4 Система управления объектной базой данных Realm

Realm – это нативная NoSQL [11] база данных для Android, iOS и JavaScript. Из ключевых особенностей стоит отметить zero copy, MVCC и ACID. Встроенного механизма устаревания и очистки данных нет.

Можно выделить три главные особенности, которые необходимо учитывать:

- Live Objects – все объекты, полученные из Realm, являются, по сути, прокси к базе данных. За счет этого достигается zero copy (объекты не копируются из базы);

- Transactions – все изменения привязанных объектов данных нужно проводить внутри транзакции;

- Open\Close – необходимость открытия\закрытия экземпляра базы данных.

Сравнение с другими базами данных:

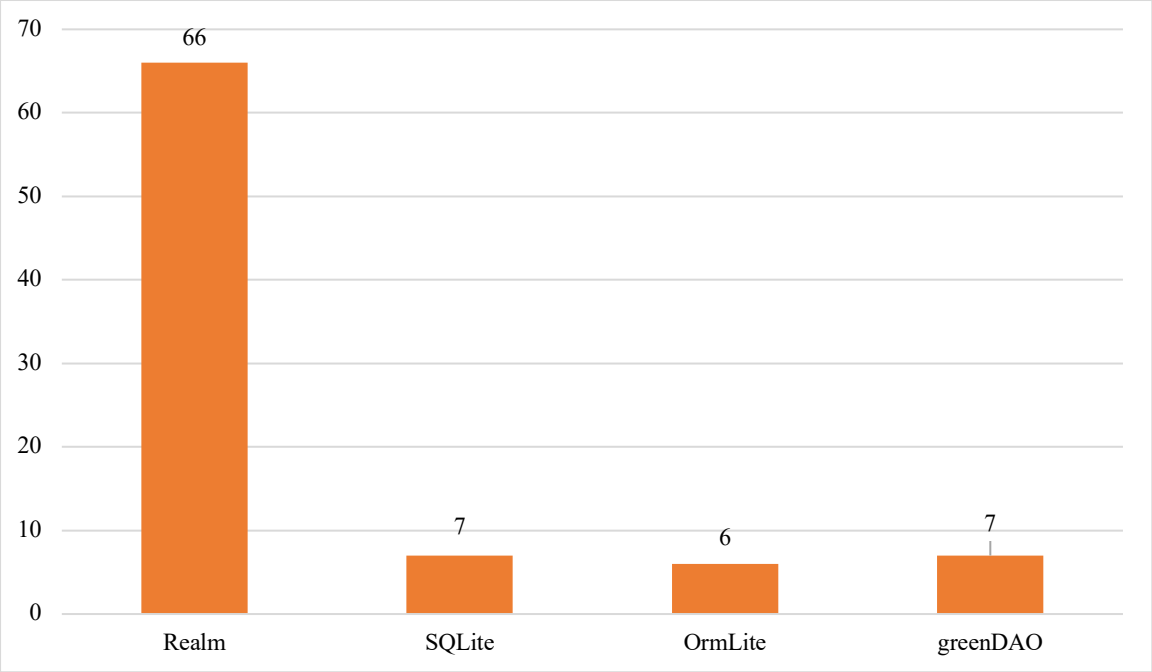


Рисунок 6 – количество запросов в секунду

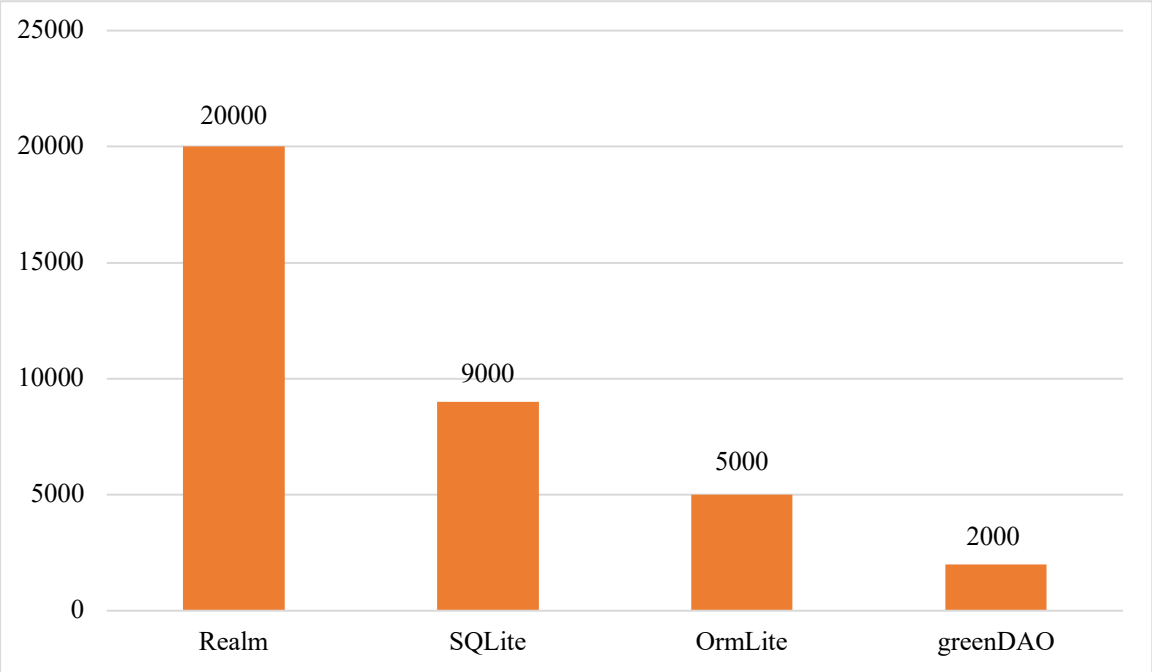


Рисунок 7 – количество записей, добавляемых в секунду

1.3.5 Программная платформа Node.js

Для того, чтобы получить полномасштабный и функциональный проект, необходимо использовать сервер, для получения и обработки пользовательских данных, а также, обработки запросов и ответов сервисов университета.

Сервер приложения «Я в СФУ» построен на базе Node.js по технологии REST. Node.js – это кроссплатформенная среда выполнения для JavaScript с открытым исходным кодом, которая работает на серверах. Платформа Node.js построена на базе JavaScript движка V8 от Google, который используется в браузере Google Chrome. Данная платформа, в основном, используется для создания веб-серверов и клиент-серверных приложений, однако сфера её применения этим не ограничивается. Движок V8 не только компилирует JavaScript во внутренний машинный код (подобно C или C++), но и делает это прозрачным образом, так что с точки зрения пользователя код ведет себя как чистый интерпретируемый язык программирования. Отсутствие отдельного шага компиляции уменьшает сложность обслуживания и развертывания [12].

Основные особенности Node.js являются:
- скорость;

- простота;
- JavaScript;
- движок V8;
- асинхронность;
- библиотеки.

1.4 Основные возможности проектируемого модуля

Разработанный программный модуль имеет следующие возможности:

- авторизация студента с помощью данных, полученных при поступлении в университет;
- отображение информация об успеваемости студента в электронных курсах;
- отображение информации из зачетной книжки студента по выбранному учебному семестру;
- отображение информации в профиле студента и списка одногруппников;
- заказ справок для студентов ИКИТа;
- связь старосты с группой путем уведомлений.

1.5 Обзор существующих решений

Идея создать данное приложение возникла из-за неудобства использования сервисов СФУ с мобильного устройства, а именно сервис «Мой СФУ» [13].

Сервис «Мой СФУ» является официальным веб-сервисом Сибирского федерального университета, но, тем не менее, мобильной версии или адаптивной веб-версии реализовано не было, а использовать веб-версию с мобильного устройства не удобно (рисунок 8).

Аналогов в виде мобильных приложений обнаружено не было.

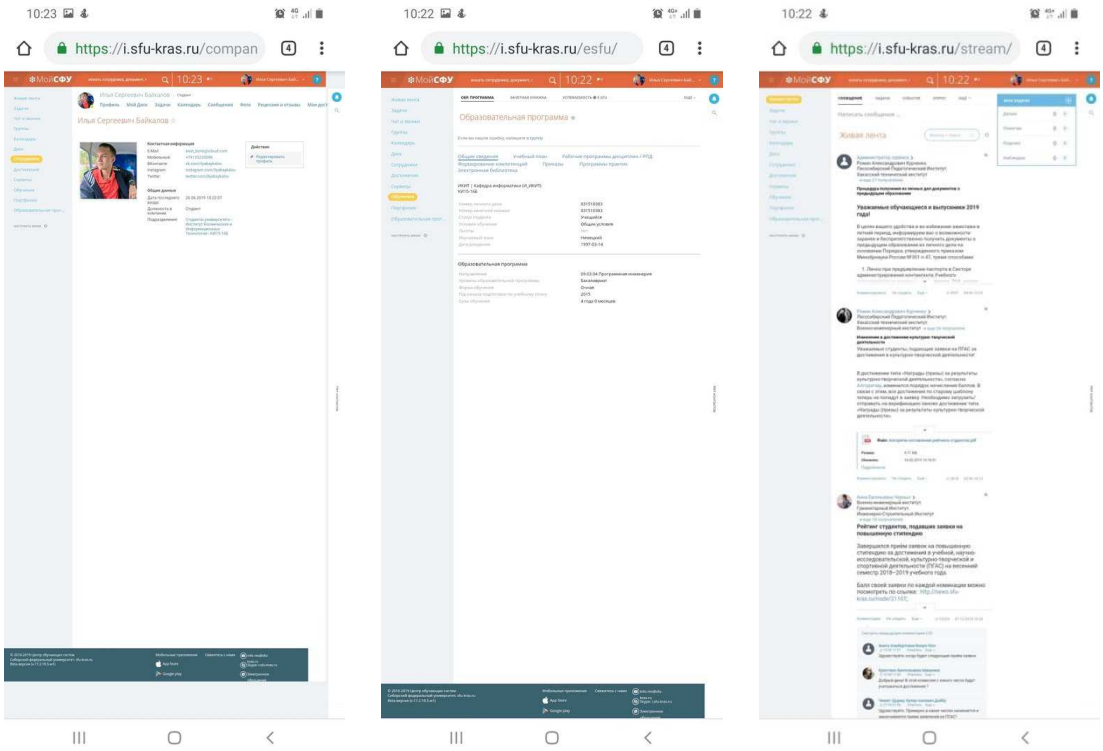


Рисунок 8 – Обзор веб-версии сервиса «Мой СФУ»

Как можно видеть на рисунке 8, веб-версия сервиса «Мой СФУ» не имеет адаптивной верстки, что приводит к тому, что пользоваться сервисом не удобно из-за того, что интерфейс даже для такого большого экрана перегружен, текст очень мелкий, а при увеличении масштаба элементы управления уходят за пределы экрана и становятся недоступными.

2 Проектирование программного продукта

Простейшая архитектура распределенного приложения, называемая архитектурой клиент-сервер, предполагает, что приложение состоит из двух частей: серверной, оказывающей услуги, и клиентской, пользующейся услугой.

В архитектуре клиент-сервер очень важно правильно разделить работу между клиентом и сервером. Можно сделать клиента, который будет отображать только результаты запроса. Это удобно для организации клиента. Его можно разместить на простейшем дешевом телефоне. Ему не нужно сложного программного обеспечения. Но тогда на сервер ложится большая нагрузка, ему приходится выполнять все запросы к источнику данных и всю обработку этих данных.

Можно, наоборот, сделать клиент, который будет выполнять всю обработку результатов запроса, а сервер, только рассылать необработанные данные клиентам. Так организован классический обмен информацией между клиентами и серверами. В этом случае для клиента требуется мощный дорогостоящий смартфон.

2.1 Архитектура приложения

Для клиент-серверного взаимодействия приложения «Я в СФУ» используется архитектурный стиль REST. REST представляет собой архитектурный стиль, который можно использовать для создания программных средств, в которых клиент может отправлять запрос на сервер [14].

Рой Филдинг, один из главных авторов HTTP-протокола, ввёл термин REST в 2000 году. Системы с поддержкой REST, называются RESTful-системами. REST имеет ряд преимуществ перед другими архитектурными стилями, а именно:

- ресурсы, возвращаемые в ответ на запрос GET, можно кэшировать множеством разных способов;

- REST дает возможность включать в каждый ресурс все состояния, необходимые для обработки конкретного запроса;

- службы RESTful не должны иметь никаких побочных эффектов, при запросе ресурса с помощью команды GET;

- команды PUT и DELETE можно использовать несколько раз, и результат будет таким же, как при их первом использовании.

Управление передачей данных основывается на протоколе передачи данных HTTP. Основными и наиболее часто используемыми HTTP методами являются POST, GET, PUT и DELETE. Ниже указаны определения основных методов из спецификаций RFC 2068 [15].

Метод POST используется для запроса, при котором адресуемый сервер принимает объект, включенный в запрос, как новое подчинение ресурса, идентифицированного запрашиваемым URL в строке запроса.

Метод GET позволяет получать любую информацию (в форме объекта), идентифицированную запрашиваемым URL. Если запрашиваемый URL обращается к процессу, производящему данные, то в качестве объекта ответа должны быть возвращены произведенные данные, а не исходный текст процесса, если сам процесс не выводит исходный текст.

Запросы с методом PUT, которые содержат объект, сохраняются под запрашиваемым URL. Если URL обращается к уже существующему ресурсу, включенный объект следует рассматривать как модифицированную версию объекта, находящегося на первоначальном сервере. Если URL не указывает на существующий ресурс, и может интерпретироваться агентом пользователя как новый ресурс для запросов, первоначальный сервер может создать ресурс с данным URL.

Метод DELETE запрашивает первоначальный сервер об удалении ресурса, идентифицируемого запрашиваемым URL.

На рисунке 9 представлена архитектура разработанного приложения.

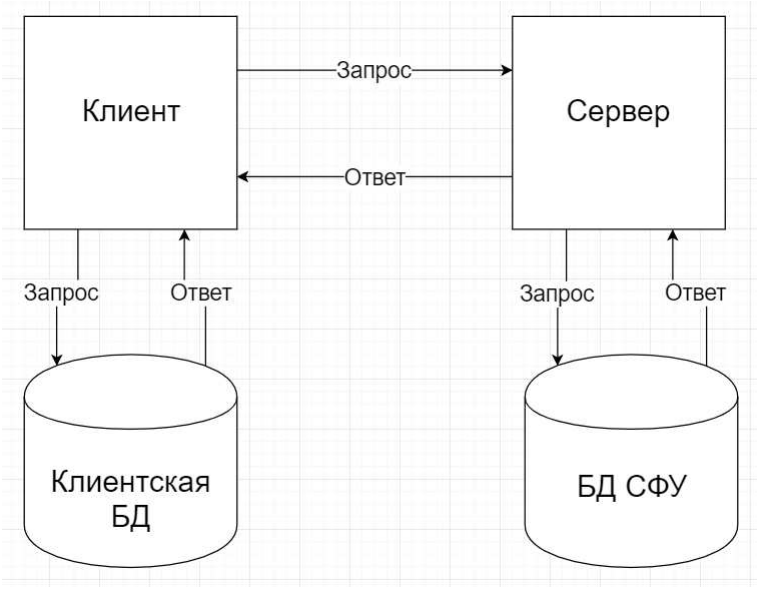


Рисунок 9 – Архитектура «клиент-сервер» приложения «Я в СФУ»

Клиент представляет собой мобильное приложение на базе Android ОС на языке программирования Java. Посредством использования библиотеки Retrofit 2 [16], которая упрощает взаимодействие с REST API сервера, было реализовано получение данных с сервера. Клиент отправляет запрос к серверу, разработанному на Node.js. Сервер, следуя командам клиента, парсит необходимую информацию с сервисов СФУ и возвращает информацию клиенту.

Клиент распоряжается информацией согласно определенным сценариям, либо сохраняет ее в локальную базу данных под управлением Realm, либо не хранит, загружая каждый раз с сервера по запросу пользователя.

Наличие локальной базы данных обеспечивает быстрый доступ к данным в случае необходимости, также экономит мобильный трафик, так как скачивание данных на устройство происходит один раз и далее данные подгружаются из локальной базы данных.

При разработке клиентской части использовался паттерн проектирования MVP (Model View Presenter). На рисунке 10 представлена схема использования MVP в нашем проекте.

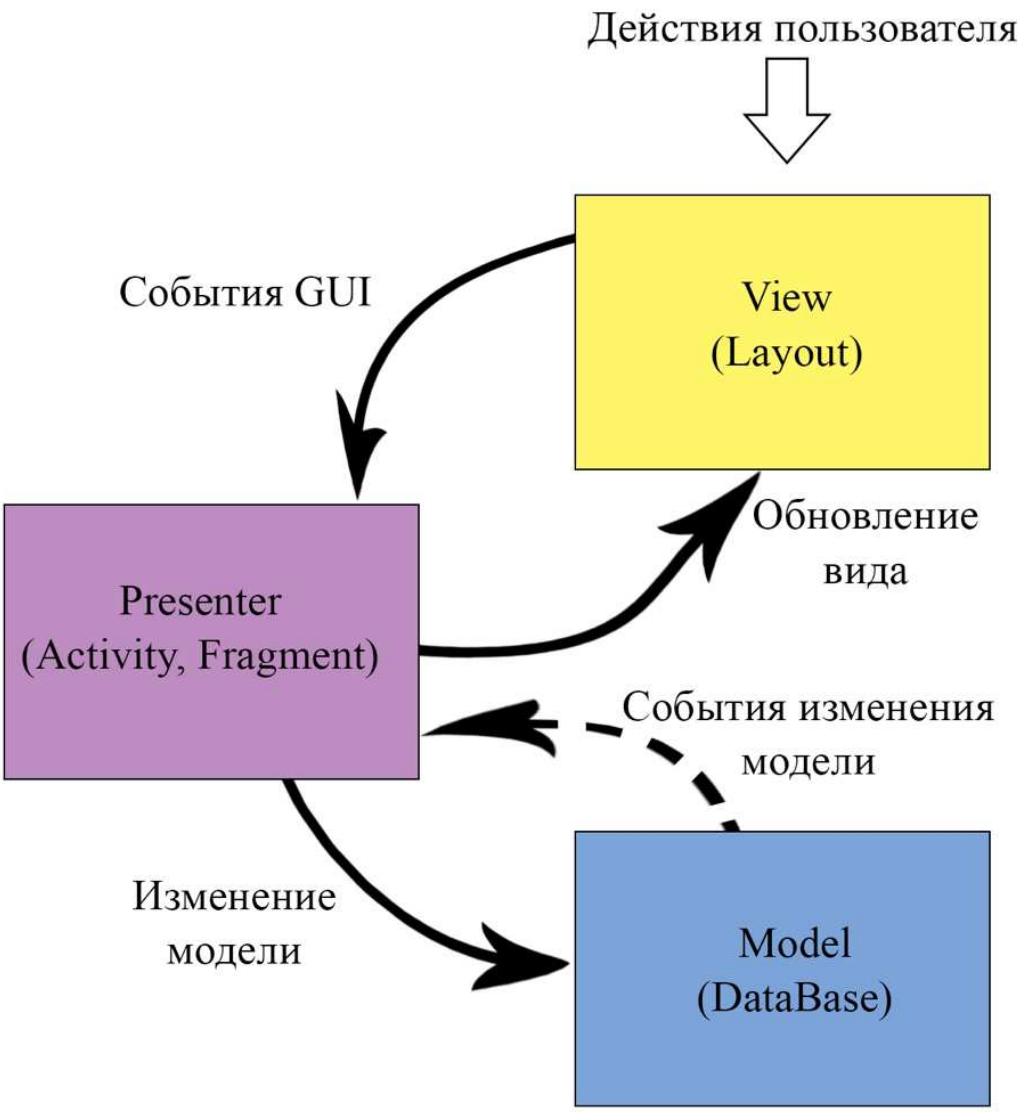


Рисунок 10 – Структура паттерна Model View Presenter

MVP – шаблон проектирования UI, который был разработан для улучшения разделения ответственности в презентационной логике:

- Model – это данные пользовательского интерфейса;

- View – используется для реализации отображения данных (Модели) и маршрутизацию пользовательских команд или событий;

- Presenter – управляет Моделью и Представлением. Например, извлекает данные из модели и формирует их для отображения в Представлении.

2.2 Структура базы данных

В проекте используется база данных СФУ для получения информации и нативная NoSQL клиентская база данных Realm [17], которая является кроссплатформенной и может быть использована как для Android (Java, Kotlin), так и для других операционных систем, например, для iOS (Objective-C, Swift). Главными особенностями базы данных Realm является ее быстродействие и хорошая программная оболочка для обращения к ней. Также в Realm присутствуют такие особенности Realm объектов, как:

- получение объектов из базы очень быстрое, десериализации как таковой нет, чтение с диска происходит только при обращении к конкретному полю;

- существует требование делать все поля приватными и обращаться через геттеры;

- метод copyFromRealm позволяет получать отвязанные, полностью собранные объекты, как обычная ORM;

- в debugger все поля будут null. Для получения какого-либо значения нам нужно обращаться через геттер;

- изменения распространяются моментально в рамках одного потока;

- фильтрация объектов осуществляется по полям, причем названия полей вы указываете руками в виде строки;

- так же одной из значимых особенностей Realm является то, что таблицы в базе не связаны индексами, в случае необходимости связать таблицы, связь осуществляется по сущностям.

На рисунке 11 представлены таблицы и их связи, используемые в приложении «Я в СФУ».

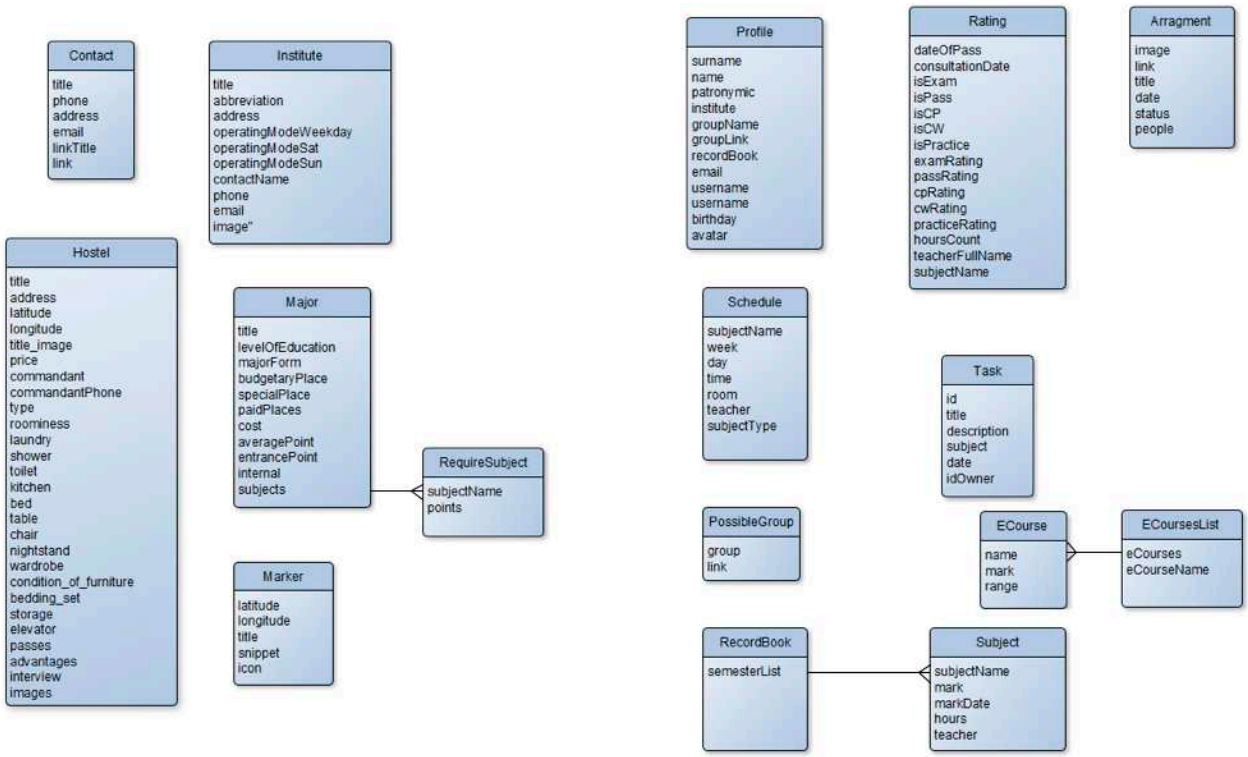


Рисунок 11 – Схема локальной базы данных Realm

В правой части рисунка 11 представлены таблицы, отвечающие за хранение информации раздела «Студент», это такие таблицы, как:

- Profile, которая отвечает за хранения информации о профиле студента и одногруппников;

- RecordBook, которая содержит информацию по зачетной книжке студента;

- Subject, которая отвечает за хранение данных об учебных дисциплинах;

- ECourse, которая содержит данные об электронных курсах;

- Rating, которая содержит информацию об оценках студента.

Наличие локальной базы данных обеспечивает быстрый доступ к данным в случае необходимости, также экономит мобильный трафик, так как скачивание данных на устройство происходит один раз и далее данные подгружаются из локальной базы данных на устройстве пользователя или по требованию пользователя.

3 Описание работы программного продукта

3.1 Серверная часть приложения

Как было сказано ранее, для организации передачи данных используется архитектура REST. В модуле «Студент» для передачи данных между сервером и клиентом используются следующие POST запросы:

- getAuthProfile, этот запрос позволяет студенту пройти авторизацию по своему логину и паролю;
- getClassmates, данный запрос отвечает за получение списка одногруппников;
- getRecordBook, данный запрос позволяет загрузить данные из зачетной книжки;
- getProgress, данный запрос позволяет получить успеваемость в электронных курсах;
- getSemesters, данный запрос позволяет получить количество учебных семестров;
- getRating, данный запрос позволяет получить оценки по учебным дисциплинам;
- getOrder, данный запрос позволяет получить историю заказанных ранее справок;
- addOrder, данный запрос позволяет заказать справку в институте.

Получение информации с сервиса «Мой СФУ» происходит путем его парсинга (web scraping) с помощью библиотеки Selenium. Парсер собирает данные со страницы студента, структурирует их и отправляет на клиент в соответствии с моделью данных. Функция, отвечающая за парсинг страницы профиля, изображена на рисунке 12. Алгоритм работы парсинга простой, сначала необходимо прогрузить нужную страницу по ее URL адресу, далее войти в личный кабинет студента и получить все данные. Далее полученные данные передаются на клиент в формате JSON.

profileQuery.js Открыть в программе «WebStorm»

```
async function getProfile(req, res) {
    let profile = new Profile();

    let username = req.body.username;
    let password = req.body.password;
    profile.setUsername = username;
    profile.setPassword = password;

    let driver = new Webdriver.Builder()
        .forBrowser('chrome')
        .setChromeOptions(options)
        .build();

    driver.get('https://i.sfu-kras.ru');

    let loginInput = driver.findElement(Webdriver.By.name("USER_LOGIN"));
    let passInput = driver.findElement(Webdriver.By.name("USER_PASSWORD"));

    await loginInput.sendKeys(profile.getUsername);
    await passInput.sendKeys(profile.getPassword);

    await driver.findElement(Webdriver.By.className("login-btn")).click();

    driver.get('https://i.sfu-kras.ru/company/personal/user/');

    let fullName = (await driver.findElement(Webdriver.By.css(".pagetitle-item")).getText()).split(" ");
    profile.setName = fullName[0];
    profile.setPatronymic = fullName[1];
    profile.setSurname = fullName[2];

    // TODO: добавить проверку на элементы
    try {
        for (let i = 1; true; i++) {
            if(await driver.findElement(Webdriver.By.css("#workarea-content > div > div > div.user-profile-block-wrap-
cont > table > tbody > tr:nth-child(" + i + ") > td:nth-child(1)")).getText() === "E-Mail:"){
                profile.setEmail = await driver.findElement(Webdriver.By.css("#workarea-content > div > div > div.user-
profile-block-wrap-cont > table > tbody > tr:nth-child(" + i + ") > td:nth-child(2)")).getText();
            }

            if (await driver.findElement(Webdriver.By.css("#workarea-content > div > div > div.user-profile-block-wrap-
cont > table > tbody > tr:nth-child(" + i + ") > td:nth-child(1)")).getText() === "Подразделения:") {
                profile.setInstitute = getInstitute(await driver.findElement(Webdriver.By.css("#workarea-content > div
> div > div.user-profile-block-wrap-cont > table > tbody > tr:nth-child(" + i + ") > td:nth-child(2) > span:nth-
child(2)")).getText());
                profile.setGroupName = await driver.findElement(Webdriver.By.css("#workarea-content > div > div >
div.user-profile-block-wrap-cont > table > tbody > tr:nth-child(" + i + ") > td:nth-child(2) > span:nth-
child(3)")).getText();
                profile.setClassmatesLink = await driver.findElement(Webdriver.By.css("#workarea-content > div > div >
div.user-profile-block-wrap-cont > table > tbody > tr:nth-child(" + i + ") > td:nth-child(2) > span:nth-child(3) >
a")).getAttribute("href");
            }
```

Рисунок 12 – Функция получения данных студента

Для того, чтобы структурировать все данные, полученные при парсинге и из файлов JSON, было принято решение использовать библиотеку для сетевого взаимодействия Retrofit 2 при их получении на клиенте. Единственное, что было необходимо это создать модель данных для всех HTTP запросов, такая же модель данных должны быть представлена в функциональной части на клиенте приложения. Пример модели данных представлен на рисунке 13, где представлена модель данных новостей. Остальные модели данных имеют схожую структуру.

Рисунок 13 – Модель данных новостей

Так, при использовании Retrofit 2, снижается часть рутиной работы по обеспечению клиент серверного взаимодействия приложения. Сервер берет на себя всю работу по обработке данных и отправляет их на клиент в формате JSON, где Retrofit 2 автоматических разбирает данные и конвертирует их в объекты, согласно построенной модели.

Серверная часть приложения была написана на Node.js и размещена на удаленном виртуальном сервере в сети, что дало возможность использовать мобильное приложение в любое время, так как удаленный виртуальный сервер

работает 24/7. Всё логирование ошибок и управление сервером происходит посредством VNC-консоли (рисунок 14).

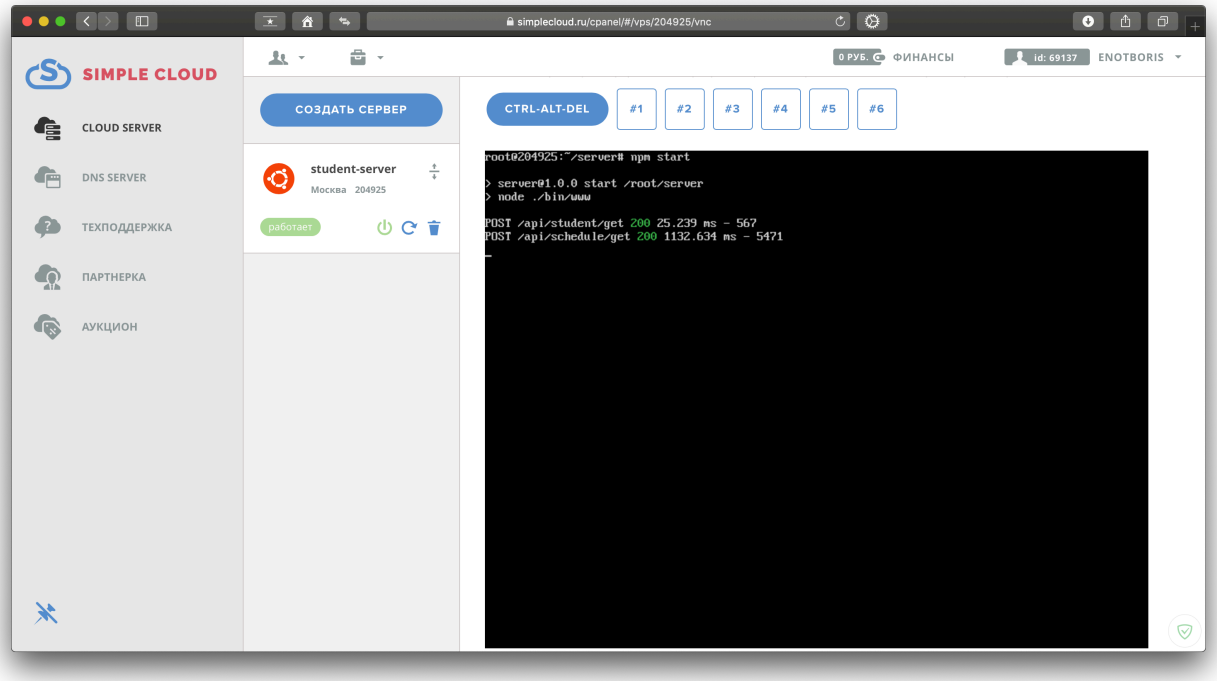


Рисунок 14 – VNC-консоль на удаленном виртуальном сервере

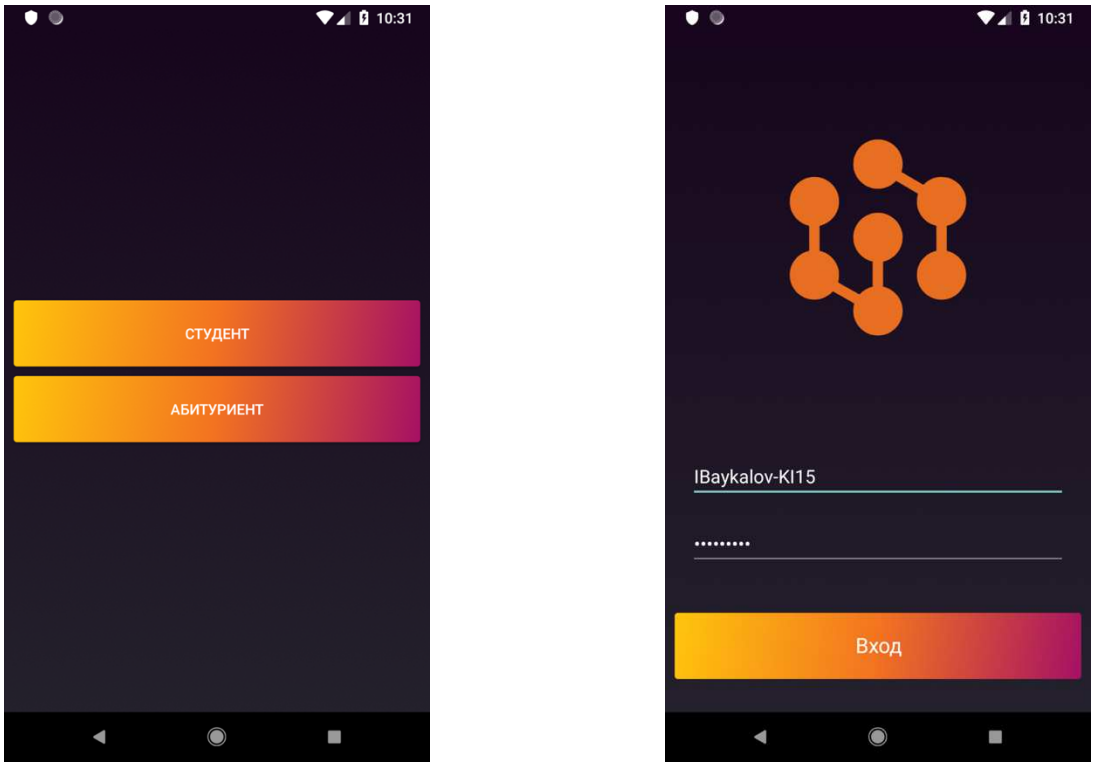
3.2 Клиентская часть приложения

3.2.1 Модуль «Авторизация»

Так как мы разрабатываем систему не только для студентов, но и для абитуриентов, было принято решение реализовать на стартовой странице приложения (рисунок 15а) выбор необходимого модуля. Разделение такого рода необходимо, так как у студента в нашем проекте предусмотрен личный кабинет с возможностью авторизации с помощью данных, которые пользователь получил при поступлении в Сибирский федеральный университет.

После выбора мобуля «Студент», пользователь попадает на страницу авторизации (рисунок 15б). Регистрация пользователя не предусматривается, так как используется существующая база данных Сибирского федерального

университета, студент использует данные для входа, которые получил при поступлении.



а б

Рисунок 15 – Стартовая страница: а) Выбор типа пользователя; б) Страница авторизации студента в приложении

После выбора раздела и успешной авторизации открывается главное меню модуля «Студент». Нижнее меню содержит такие разделы как:

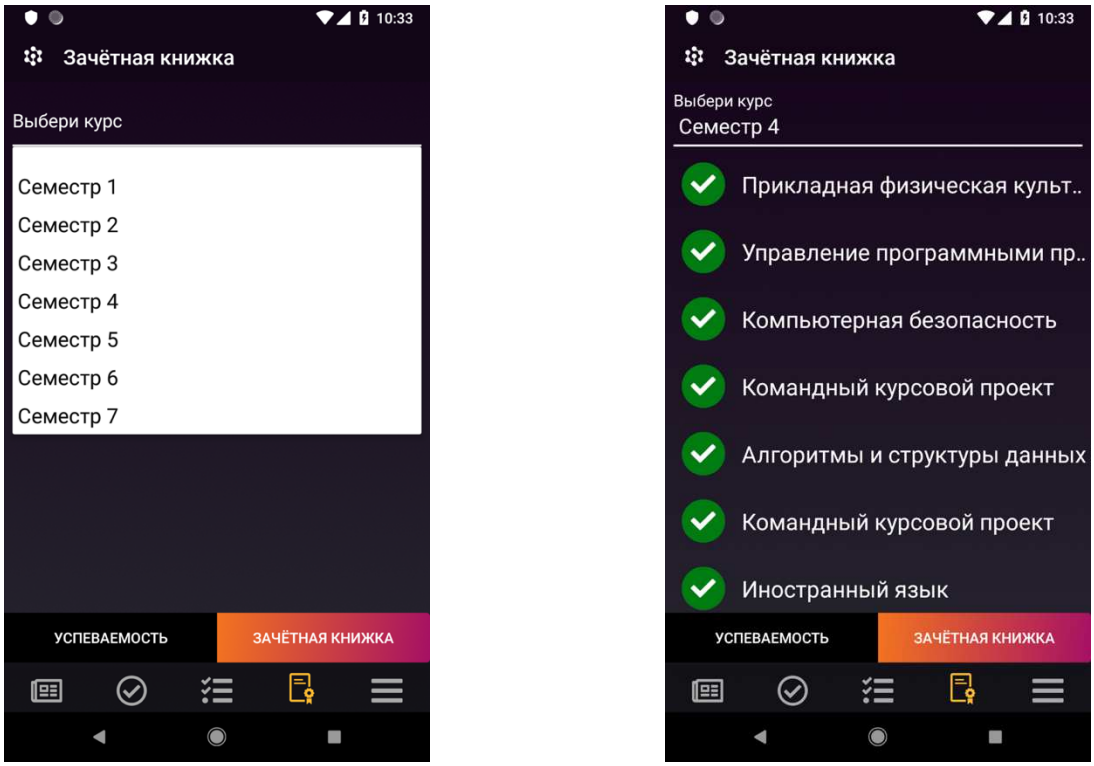
- новости;
- таскменеджер;
- расписание;
- успеваемость и зачетная книжка;
- дополнительное меню.

Дополнительное меню включает в себя:

- профиль студента и группа;
- заказ справок;
- профориентация.

3.2.2 Модуль «Зачетная книжка»

Одним из самых важных для любого студента является его зачетная книжка. Классический вариант данного документа был переработан в электронный и представлен в удобном виде в нашем приложении. Как можно увидеть на рисунке 16, студент может выбрать интересующий семестр (рисунок 16а), после чего появятся учебные дисциплины по выбранному семестру (рисунок 16б).



а б

Рисунок 16 – Электронная зачетная книжка: а) Выбор учебного семестра; б) Отображение учебных дисциплин.

Для того чтобы увидеть детали по выбранной дисциплине необходимо сделать нажатие по ней. При раскрытии деталей будут отображены преподаватель, количество учебных часов, оценка за экзамен или зачет-незачет и дата проведения экзамена или зачета (рисунок 17).

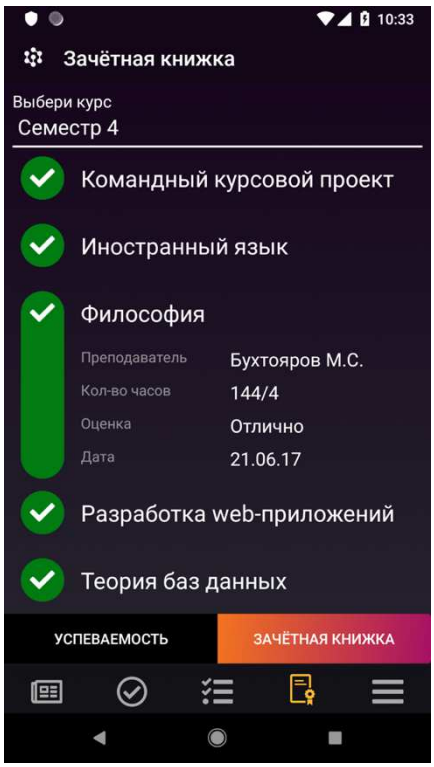
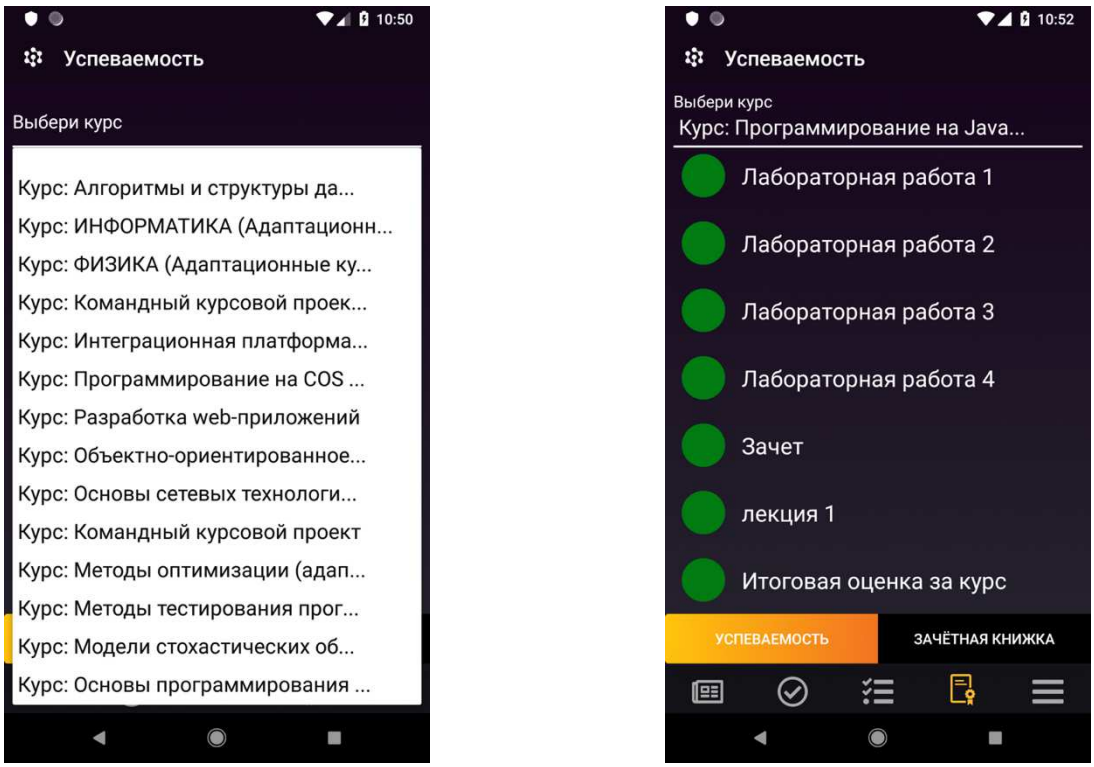


Рисунок 17 – Отображение деталей по выбранной дисциплине

3.2.3 Модуль «Успеваемость»

Данный модуль схож с предыдущим модулем и имеет схожий интерфейс. Модуль «Успеваемость» необходим, чтобы просматривать прогресс обучения в электронных курсах.

Как можно увидеть на рисунке 18а, модуль дает возможность выбрать любой электронный курс, к которому студент был подключен за время учебы и просмотреть детали, как показано на рисунке 18б. Отображение деталей по занятиям происходит по аналогии, как и в зачетной книжке, по нажатию на выбранное занятие (рисунок 19).



а б

Рисунок 18 – Успеваемость студента: а) Выбор электронного курса; б) Отображение успеваемости в зависимости от выбранного электронного курса

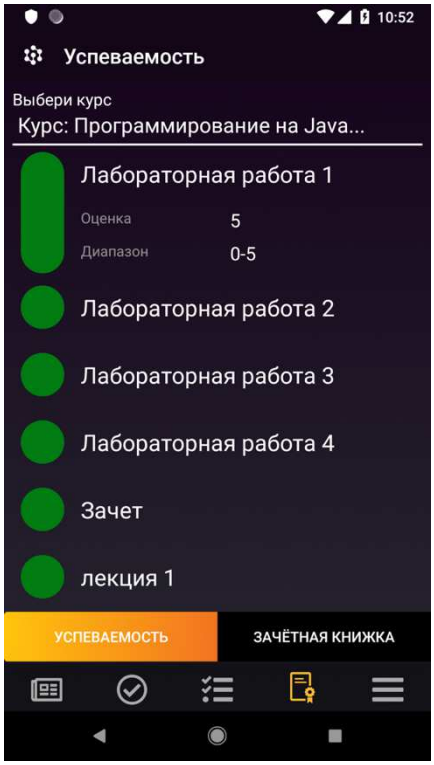


Рисунок 19 – Отображение деталей по заданиям в электронном курсе

3.2.4 Модуль «Профиль студента»

Модуль «профиль студента» (рисунок 20) является дополнительным, так как не имеет важного функционала. Он необходим для изменения или добавления информации о себе для того, чтобы одногруппники имели актуальную информацию о студенте.

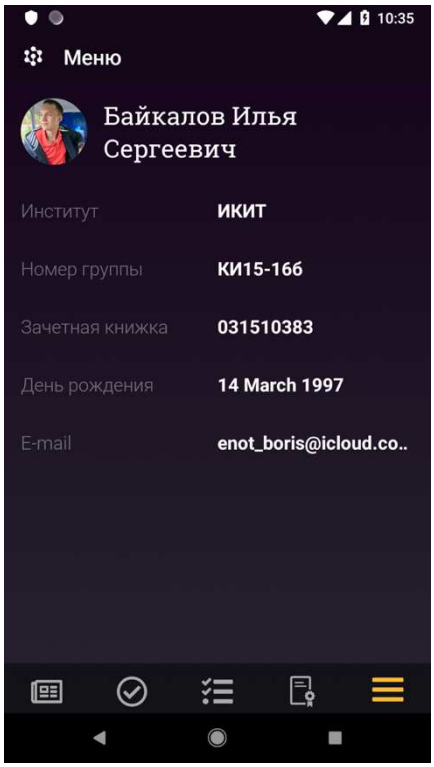
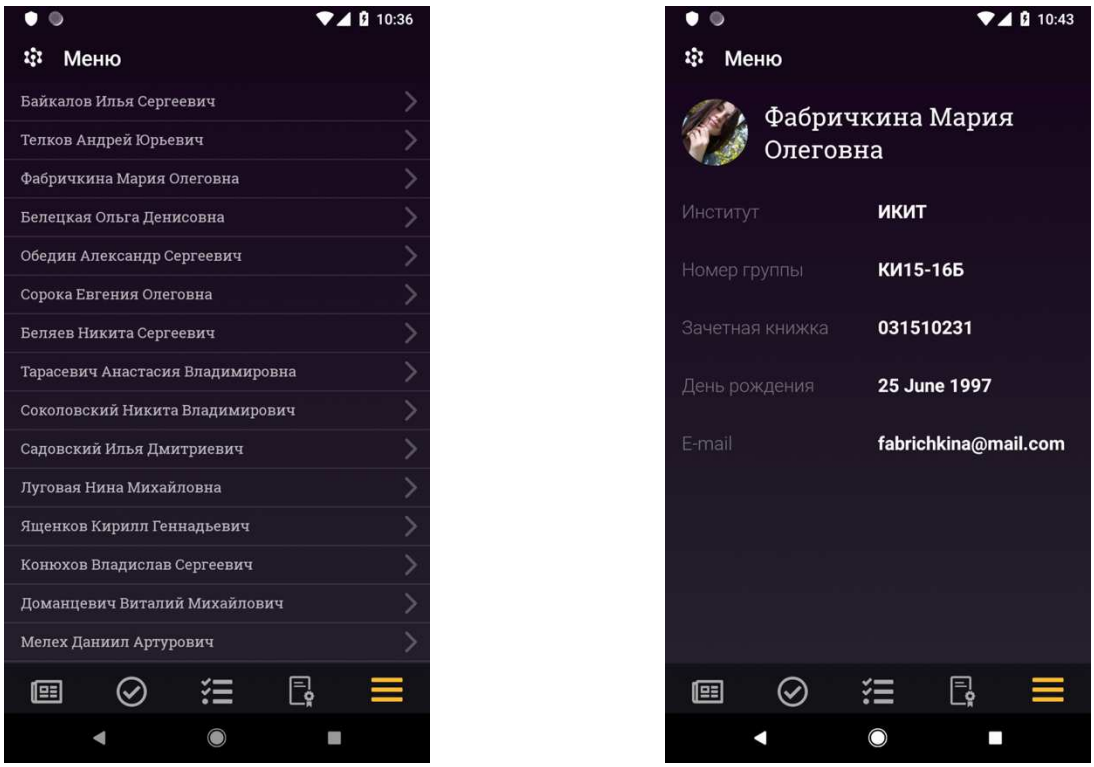


Рисунок 20 – Внешний вид профиля студента

3.2.5 Модуль «Моя группа»

Функционал модуля «Моя группа» состоит в том, чтобы выводить актуальный список одногруппников (рисунок 21а), отображать краткую информацию о студенте, такую как фамилия, имя, отчество, дата рождения и электронная почта (рисунок 21б). В приложении отображается пользовательский аватар, при условии, что он изначально был в профиле на сайте «Мой СФУ».



а б

Рисунок 21 – Модуль «Моя группа»: а) Отображение списка одногруппников; б) Профиль выбранного одногруппника

3.2.6 Модуль «Заказ справок»

Данный модуль реализован для упрощения процесса заказа справок для студентов ИКИТа. Он полностью повторяет функционал заказа справок в электронном деканате ИКИТа, но, так как данный сервис представлен в мобильном приложении, теперь есть возможность пользоваться данным сервисом с мобильных устройств.

В функционал входит заказ справки по месту требования и для пенсионного фонда, есть возможность указать необходимое количество и необходимость гербовой печати. Так же добавлена информационная справка о сроках выполнения и других типах справок (рисунок 22).

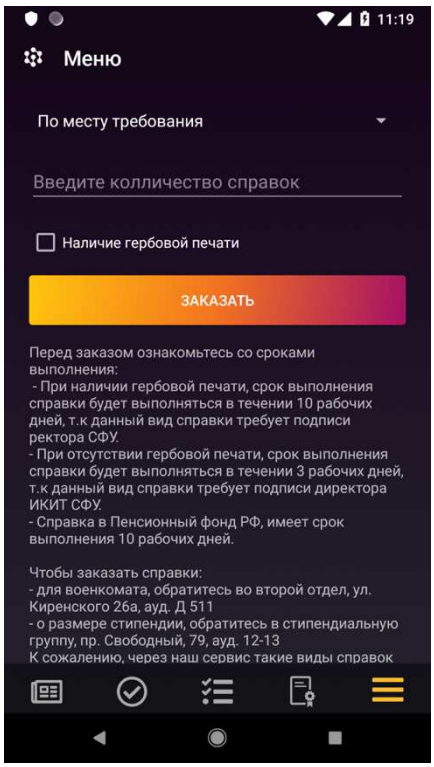


Рисунок 22 – Окно заказа справок

4 Тестирование программного продукта

4.1 Тестирование удобства использования

Тестирование удобства использования программного продукта [18] – это тестированное, с помощью которого можно проверить ориентированность приложения на пользователя. При тестировании удобства использования главной целью является проверка того, насколько легко новому пользователю понять, как работать с приложением. Иначе говоря, тестируется системная навигация всего приложения.

Тест удобства использования удостоверяется в простоте и эффективности использования продукта при использовании стандартных практик тестирования удобства использования.

Сценарии тестирования удобства использования:

- содержание окон мобильного приложения верное, без грамматических и орфографических ошибок;
- все шрифты соответствуют требованиям;
- оптимальность выбранной цветовой палитры;
- все тексты правильно выравнены;
- все поля правильно выравнены;
- между полями, колонками, рядами и сообщениями об ошибках оставлено достаточно свободного места;
- все кнопки должны иметь стандартный формат и размер;
- неактивные поля должны быть серыми;
- адаптация приложения при разных разрешениях экрана;
- панель скролла должна появляться только тогда, когда она требуется;
- заголовок должен отображаться на каждой странице;
- данные в выпадающих списках не обрезаются из-за размеров поля.

Для тестирования программного продукта была собрана фокус-группа независимых пользователей, каждый из которых дал свою оценку по

вышеупомянутым пунктам по десятибалльной шкале, чтобы более точно проанализировать то, как пользователи оценивают программный продукт. Результаты тестирования представлены в таблице 1.

Таблица 1 – Тестирование на удобство использования

Участник	**Цвета**	**Шрифты**	**Предупре-ждающие сообщения**	**Ссылки и изображения**	**Заголо-вок**	**Удобство кнопок**	**Удобство полей и форм**	**Нави-гация**
Среднее значение	8,25	7,75	7,625	7,875	8,25	8,5	7,75	8,125

Основываясь на полученных результатах тестирования, можно сделать вывод, что разработанное приложение является достаточно удобным и проработанным с точки зрения взаимодействия пользователя с интерфейсом приложения.

Наряду с тестированием удобства использования, были проведены проверки на содержание информационного наполнения приложения, а именно: наличие грамматических и орфографических ошибок, информативность, структурированность и логическую связанность информации.

По полученным результатам тестирования, приложение было доработано, устранены замечания пользователей из фокус-группы. Исправлены недостатки по расположению частей интерфейса, недочеты информационного характера.

Таким образом, можно сказать, что тестирование удобство использования было проведено успешно и помогло устранить существующие на тот момент недостатки программного продукта.

4.2 Функциональное тестирование

Функциональное тестирование программного продукта подразумевает под собой тестирование взаимодействий с пользователями, тестирование транзакций

сервера с клиентом, а также проверку продукта на соответствие функциональной сертификации.

Наиболее важным моментом функционального тестирования, является проверка вывода информации и наличие информативных сообщений о возникающих ошибках при некорректном поведении пользователя.

В модуле «Организация учебного процесса» есть несколько окон, на которых происходит отправка данных на сервер и получения данных в качестве ответа.

При переходе в модуль «зачетная книжка» на сервер отправляется id студента, по которому возвращается список семестров. Далее, при выборе семестра на сервер отправляется id семестра, по которому возвращается список учебных дисциплин.

В модуле «Успеваемость» процесс получения учебных курсов и занятий происходит по аналогии – сначала возвращается список курсов, после выбора электронного курса возвращается список учебных занятий.

4.3 Тестирование совместимости

Тестирование совместимости [19] используется, чтобы убедиться, что приложение совместимо с другими элементами системы, в которой оно работает, например, операционными системами или железом.

Цель тестирования совместимости – оценка того, насколько хорошо ПО работает под определенной версией Android OS, с другим ПО или железом.

Сценарии тестирования совместимости:

- убедиться, что приложение достаточно адаптировано под разные разрешения экрана, предусмотренных ТЗ;
- убедиться, что картинки корректно отображаются на разных разрешениях экрана, предусмотренных ТЗ;
- убедиться, что шрифты, верно, отображаются на разных разрешениях экрана, предусмотренных ТЗ;

- убедитесь, что приложение корректно работает на разных версиях Android OS, предусмотренных ТЗ.

В таблице 2 представлены результаты тестирования различных элементов на Android OS под перечисленные выше требования.

Таблица 2 – Результаты тестирования совместимости

Версия Android ОС	**Lollipop (5.0)**	**Marshmallow (6.0)**	**Nougat (7.0)**	**Oreo (8.1)**
Выбор типа пользователя	корректно	корректно	корректно	корректно
Просмотр новостей	корректно	корректно	корректно	корректно
Создание задачи	корректно	корректно	корректно	корректно
Просмотр списка задач	корректно	корректно	корректно	корректно
Просмотр расписания	корректно	корректно	корректно	корректно
Просмотр профориента-ционных событий	корректно	корректно	корректно	корректно

Долгое время производители считали соотношение сторон 16:9 единственно приемлемым, но с увеличение длинны диагонали, начиная от 5,5 дюйма, устройство с соотношением 16:9 кажется громоздким, поэтому в 2017 [20] появились первые смартфоны с соотношением экрана 18:9, которые продолжают набирать популярность благодаря своей эргономичности. В таблице 3 показаны результаты тестирования на экранах с различным соотношением сторон.

Таблица 3 – Результаты тестирования отображения

Соотношение сторон экрана	**16:9**	**18:9**
Переходы между страницами	корректно	корректно
Отображения надписей, кнопок, полей	корректно	корректно
Корректные расстояния между элементами	корректно	корректно
Верное отображение текста	корректно	корректно

Таким образом, по результатам тестирования на совместимость приложения для различных соотношений сторон экрана и разных версий

Android OS можно сделать вывод, что мобильное приложение корректно отображается, а все функции выполняются верно, независимо от того, с каким соотношением сторон экранов необходимо иметь дело.

ЗАКЛЮЧЕНИЕ

По результатам работы был реализован модуль «Организация учебного процесса» и, совместно с усилиями остальными разработчиками, было получено стабильно работающее клиент-серверное мобильное приложение «Я в СФУ», состоящее из трех модулей и объединяющее функционал для студента и абитуриента в одно информационное пространство, что делает его уникальным в подобного рода приложениях. Весь функционал, определенный в техническом задании, был реализован и протестирован, все поставленные задачи были решены и цель достигнута.

Было получено свидетельство о государственной регистрации программы для ЭВМ №2019615282 от 23.04.2019 г., представленное в приложении А.

В данном мобильном приложении заинтересованы как довузовское управление Сибирского федерального университета, так и учебный отдел. В настоящее время идут переговоры о дальнейшем развитии и интеграции приложения в информационное пространство СФУ.

В дальнейшем в модуле «Организация учебного процесса» планируются некоторые улучшения, в частности, реализация единой информационной системы для Сибирского федерального университета, внедрение нового функционала в уже существующие сервисы СФУ, замена всех существующих порталов университета и всех институтов единой информационной системой.

Также, в будущем, планируется проектирование и реализация третьего модуля «Преподаватель» для приложения «Я в СФУ», который будет включать в себя функционал, полезный для преподавателей, например, реестр научных трудов преподавателя с возможностью подачи его в различные конкурсы/гранты, расписание преподавателя и другой необходимый функционал. Кроме того, планируется разработать приложение для системы iOS и внедрить возможность персонализации интерфейса приложения.

СПИСОК СОКРАЩЕНИЙ

API – Application Programming Interface

HTML – HyperText Markup Language

HTTP – HyperText Transfer Protocol

JSON – JavaScript Object Notation

ORM – Object-Relational Mapping

OS – Operating System

REST – Representational state transfer

URL – Uniform Resource Locator

XML – eXtensible Markup Language

ВУЗ – Высшее учебное заведение

ПО – программное обеспеченье

СФУ – Сибирский федеральный университет

ТЗ – Техническое задание

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Вигерс, К. И. Разработка требований к программному обеспечению.: Пер. с англ. / К. И. Вигерс. – М.: Издательско-торговый дом «Русская Редакция», 2004. – 576 с.

2. Леффингуэлл, Д. Принципы работы с требованиями к программному обеспечению. Унифицированный подход.: Пер. с англ. / Д. Леффингуэлл, Д. Уидриг. – М.: Издательский дом «Вильямс», 2002. – 448с.

3. Desktop vs Mobile vs Tablet Market Share Worldwide [Электронный ресурс]: StatCounter предоставляет статистические данные по выборке, превышающей 10 миллиардов просмотров страниц в месяц, собранных со всей сети StatCounter из более чем 2 миллионов веб-сайтов. // StatCounter GlobalStats. – Режим доступа: http://gs.statcounter.com/platform-market-share/desktop-mobile-tablet/worldwide/2018

4. Streaming delays mentally taxing for smartphone users: Ericsson Mobility Report [Электронный ресурс]: Пресс-релиз от 17.02.2016. // Ericsson. – Режим доступа: https://www.ericsson.com/en/press-releases/2016/2/streaming-delays-mentally-taxing-for-smartphone-users-ericsson-mobility-report

5. Mobile Operating System Market Share Russian Federation [Электронный ресурс]: StatCounter предоставляет статистические данные по выборке, превышающей 10 миллиардов просмотров страниц в месяц, собранных со всей сети StatCounter из более чем 2 миллионов веб-сайтов. // StatCounter GlobalStats. – Режим доступа: http://gs.statcounter.com/os-market-share/mobile/russian-federation

6. Distribution dashboard [Электронный ресурс]: Сайт посвященный Android Platform. // Android Developers. – Режим доступа: https://developer.android.com/about/dashboards

7. IntelliJ IDEA [Электронный ресурс]: Продукты и разработки. // Jet Brains. – Режим доступа: https://jetbrains.ru/products/idea/

8. WebStorm – умная IDE для JavaScript-разработчиков [Электронный ресурс]: Продукты и разработки. // Jet Brains. – Режим доступа: https://jetbrains.ru/products/webstorm/

9. Сол, К. XML: структурирование данных для Web: Введение [Электронный ресурс] / К. Сол // Журнал BPM World. – 1998. – Режим доступа: https://iso.ru/ru/press-center/journal/1715.phtml

10. Светлова, И. История языка программирования Java [Электронный ресурс] / И. Светлова // Oracle Patches. – 2018. – Режим доступа: https://oracle-patches.com/coding/3415-история-языка-программирования-java

11. NoSQL базы данных: понимаем суть [Электронный ресурс]. – Режим доступа: https://habr.com/ru/post/152477/ (дата обращения: 15.06.2019)

12. Браун, И. Веб-разработка с применением Node и Express. Полноценное использование стека JavaScript. / И. Браун. – СПб.: Питер, 2017. – 336 с.

13. Сервис «Мой СФУ» [Электронный ресурс] // Сибирский федеральный университет. – Режим доступа: https://i.sfu-kras.ru (дата обращения: 17.06.2019)

14. Фландерс, Д. Введение в службы RESTful с использованием WCF [Электронный ресурс] / Д. Фландерс // MSDN Magazine. – 2009. – Режим доступа: https://msdn.microsoft.com/ru-ru/magazine/dd315413.aspx

15. Филдинг, Р. RFC 2068 — Протокол Передачи Гипертекста - HTTP/1.1 [Электронный ресурс] / Р. Филдинг, Дж. Геттис, Дж. Могул, Г. Фристик, Т. Бернерс-Ли // RFC – 1997. – Режим доступа: https://rfc2.ru/2068.rfc/original

16. Retrofit 2 – A type-safe HTTP client for Android and Java [Электронный ресурс]. – Режим доступа: https://square.github.io/retrofit/ (дата обращения: 19.06.2019)

17. Realm — кроссплатформенная мобильная база данных [Электронный ресурс]: Блог компании Jetruby. // Jetruby. – Режим доступа: https://jetruby.com/ru/blog/realm-mobilnaya-baza-dannyh/

18. Что такое usability и зачем нужны usability тестирования? [Электронный ресурс]. – Режим доступа: https://habr.com/ru/sandbox/122757/ (дата обращения: 24.06.2019)

19. 10 software testing trends to watch out for in 2019 [Электронный ресурс]. – Режим доступа: https://reqtest.com/testing-blog/software-testing-trends-2019/ (дата обращения: 24.06.2019)

20. Первые в мире смартфоны с экраном 18:9 – что в них такого? [Электронный ресурс]: Новости высоких технологий hi-tech Mail.ru. // Hi-tech – Режим доступа: https://hi-tech.mail.ru/review/lg-q6/

ПРИЛОЖЕНИЕ А

Свидетельство о регистрации программы для ЭВМ

РОССИЙСКАЯ ФЕДЕРАЦИЯ

СВИДЕТЕЛЬСТВО

о государственной регистрации программы для ЭВМ

№ 2019615282

Мобильное приложение «Личный кабинет студента СФУ». Модуль «Организация учебного процесса»

Правообладатель: ***Байкалов Илья Сергеевич (RU)***

Автор: ***Байкалов Илья Сергеевич (RU)***

Заявка № **2019614095**
Дата поступления **12 апреля 2019 г.**
Дата государственной регистрации
в Реестре программ для ЭВМ ***23 апреля 2019 г.***

Руководитель Федеральной службы
по интеллектуальной собственности

Г.П. Ивлиев

Рисунок А. 1 – Свидетельство о государственной регистрации программы для ЭВМ

Федеральное государственное автономное
образовательное учреждение
высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт космических и информационных технологий

институт

Кафедра «Информатика»

кафедра

УТВЕРЖДАЮ
Заведующий кафедрой

_____ А. С. Кузнецов

подпись инициалы, фамилия

«05» 07 2019 г.

БАКАЛАВРСКАЯ РАБОТА

09.03.04 Программная инженерия

Код – наименование направления

Разработка модуля «Организация учебного процесса» для мобильного приложения «Я в СФУ»

тема

Руководитель _____ 03.07.19 доцент, к. т. н. А. В. Хныкин

подпись, дата должность, ученая степень инициалы, фамилия

Выпускник _____ 03.07.19 И. С. Байкалов

подпись, дата инициалы, фамилия

Нормоконтролер _____ 05.07.19 доцент, к. т. н. О. А. Антамошкин

подпись, дата должность, ученая степень инициалы, фамилия

Красноярск 2019